Homework 3

CMSY-217, Spring 2014

Upload your solution to the Canvas course website as a zip archive file prior to the start of class on Thursday, March 27.

1. Write a Java application called `LoanCalculator` which is a subclass of `JFrame` and implements an `ActionListener` interface. In addition to the `main` and `actionPerformed` methods, the `LoanCalculator` class will have one instance variable of type `UserInterface` and a no-argument constructor which initializes the `UserInterface` and adds it to the content pane of the `LoanCalculator`.

2. Write a Java class called `UserInterface` which is a subclass of `JPanel` and has the Swing components as shown in the figure below. You may use the GUI Builder in the NetBeans IDE to create this class or code it by hand using the techniques presented in Chapter 14 of the textbook.

3. In the `UserInterface` class, provide getter methods which return a `String` for the three `JTextField` objects (loan amount, interest rate, and term) and a getter method which returns an `int` for the `JComboBox` (payment plan). Also provide setter methods which take a String for the two `JLabel` objects that will display the calculated monthly payment and total payout.

4. Write a class called `Validator` which has a four-argument constructor to set its instance variables - three String objects (loan amount, interest rate, and term) and an `int` (payment plan). When the user presses the compute button, a `Validator` object should be created which has a method to ensure that the loan parameters satisfy the following constraints

   (a) A loan amount in dollars which must be a positive value and need not be an integral value

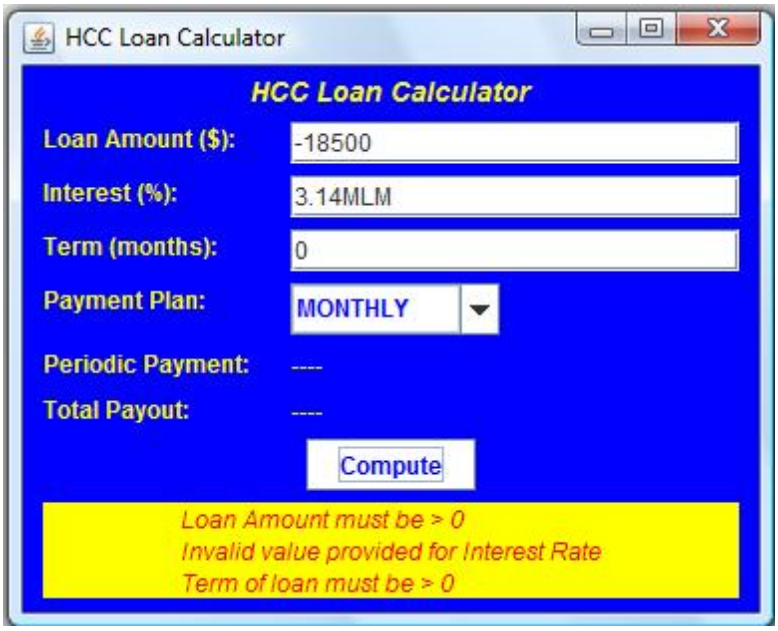
   (b) An annual interest rate for the loan which must be a decimal between 0 and 100 (inclusive) so a four and a half percent interest rate will be input as 4.5 and not 0.045.

   (c) The number of payments to be made over the course of a year which will be limited to the following options:

      i. Monthly - 12 payments per year
      ii. Quarterly - 4 payments per year
      iii. Biannually - 2 payments per year
      iv. Annually - 1 payment per year

   (d) The term of the loan in months which must be a positive integer

   If the parameter(s) are determined to be invalid, provide a useful error message for each invalid input as shown in the figure below.



5. Provide getter methods in the `Validator` class which return the loan amount, interest rate, number of payments per year, and term of the loan as a `double`.

6. Write a class called `Formula` which has methods to compute and return the periodic payment and total payout as a formatted `String` with two decimal places. Use the following equations to compute the periodic payment and total payout when given valid input parameters from the `Validator` object.

```
double p = loanAmount;
double r = interestRate;
double t = term;
double f = paymentPlan;

double m = 12 / f;
double i = Math.pow(1 + r/1200, m) - 1;
double v = 1 / (1 + i);
double n = t * f / 12;
double a = (1 - Math.pow(v,n)) / i;

double periodicPayment = p / a;
double totalPayout = n * periodicPayment;
```

7. Compile your project and package it as an executable JAR file. Write a JNLP file which allows you to deploy your project as a Java Web Start application. For more details on Java Web Start, please visit the following URL.
http://docs.oracle.com/javase/tutorial/deployment/webstart/deploying.html

8. Reimplement your project as an applet by copying the `LoanCalculator` class as `LoanCalculatorApplet` and making it a subclass of JApplet instead of JFrame. For more details on Applets, please visit the following URL. http://docs.oracle.com/javase/tutorial/uiswing/components/applet.html